Chapter-4:

Arrays, Tables and Matrix

Dulal Acharjee

Arrays are variables with same name but different subscript values. Advantages of declaring arrays are that by a single name of variable we can get unlimited number of variables to input data. If we say ***x[4]***, there are ***x[0], x[1], x[2]*** and ***x[3]*** four variables; all variable names are x but subscripts are different. In that way, if we need to input 4 names of students, we can declare an String type of array as ***fname[3]*** which creates ***fname[0], fname[1], fname[2]*** and ***fname[3]***. In JAVA, subscript starts from zero and ends at (n-1) number and written as X[2], abc[5] or sal[10] in these ways. Advantage of declaring arrays is that instead of creating many variables, it is possible to create many variables only by changing the subscript number. If we need to input 1000 students name, it is impractical to define 1000 variables. But if it is declared as ***String fname[999]*** or ***String[999] fname;*** then 1000 variables are declared by one statement. Later it will be discussed that for uses of 1000 names we need to create 1000 rooms within the physical memory of the computer.

In the same way, numerical variables can be created in different ways as:

```
int [ ] x;
float y[ ]; // third bracket symbols can be given after data type of after the variable.
double z[];  //etc.
```

A Table can be of one column or of more columns as seen in excel sheet. In a table, vertical lines are known as column and horizontal lines are known as rows. Each of data in Excel sheet, like a table, has an address which is composed with its Row and Column values. We need to learn this concept as because, we need to read or write data from/to a data sheet. Say, salaries of 100 employees are put in a table; in that case, it is represented as salary[100]. This can be represented by salary[r] while r is between 0 and 99 as 0<= r <=99. In a program using a loop, if we increase the value if r by one, we can access all salary amount one by one.

Concept of array is described with the help of a figure:

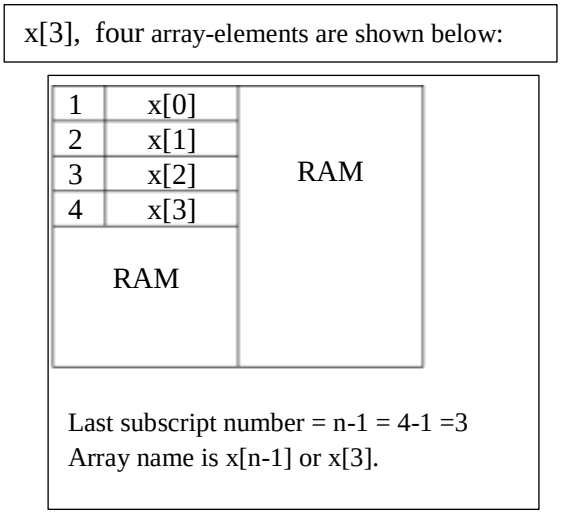

Fig 3.1. Shows an array with its subscript values how it takes position within RAM.

One Dimensional Array:

In mathematical notations, one dimensional array is written in different ways as:

X^n where n = 0…(n-1); it means there are 'n' number of variables as X^0,X^1,X^2…X^{n-1}. If we write as, X^3, it has four variables as X^0, X^1, X^2. Here, it is known that it is one dimensional array, because, it has one variable as subscript. If we put the subscript as X_n, that is also known as one dimensional array.
P_i where i = 1…m(a upper bound of i); it has also p_1, p_2, p_3…p_m variables of one dimension.
In these two example, both are representations of one dimensional arrays, only notations are different positions as superscript and subscript.
In programming concept it is call array of a variable. In Java an one dimensional arra is declared as:

```
int[] x;  or  int x[];
```

After declaring, we need to initialize or link with an memory address of the computer memory as it is done by the statement: *x= new int[3];*it has a meaning that keep memory spaces for holding 3 integer numbers, that is, 3*4=12 bytes will be allocated and the address of the first memory block will be assigned into x.

Array is the facility to create large numbers of variables by one name- only changing its subscript numbers. If it is required to input 1000 numbers, then it is not possible to declare 1000 variables like, x1,x2,x3…,x1000; if it is required to input large numbers of city names of the World, using facility of array declaration, to declare 1000 city names can be done in this way as:

```
String[] city;
city=new String[1000];
```

These two lines can be written as a single like as:

```
String city = new String[1000];
```

For numerical arrays of int, byte, short, float etc. can be data type as we have described before.

```
  int[ ] x;
 or ,   int x[];  // has same meaning
  x = new int[4]; //initialization
or ,  int x = new int[4];// same meaning.
```

First statement declares array of *x* as data type of integer. But, length of the array or how many numbers can be inserted into *x* that is not mentioned by this statement. For that reason, second line is declared as: '*x = new int[4]*' which assigns four variables within the memory of the computer as: x[0], x[1], x[2] and x[3]. Number starts from zero, so, 0, 1, 2, and 3 are the four numbers used to generate arrays of x[4]. Subscript [4] means that total four number of arrays will be generated and the highest subscript value will be (4-1) =3. Starting from 0 to 3 there are four numbers. But, never it generate x[4]; there is no existence of x[4], compiler of JAVA has not created this array or the fifth array, it has generated upto fourth array , i.e, x[0], x[1], x[2] and x[3].

Problem: wap to declare an *int* array of 10 elements, make a while loop to assign array value by counter number, and then print the value of the array.

```
//ArrayDemo.java : one dimensional array, assign and print 1 to 10 values using while loop
import java.util.*;
class ArrayDemo
{
  public static void main(String args[ ])
  {
          int[ ] x = new int[10];
          int i=0, n=9;
          //start a loop to continue upto n times
          while(i<=n)
          {
            x[i]=i;
             System.out.println("The no is:"+  x[i]);
             i++; // i=i+1, increases by one
          }
   }
}
```

```
OUTPUT
The no is:0
The no is:1
The no is:2
The no is:3
The no is:4
the no is:5
the no is:6
the no is:7
```

```
the no is:8
the no is:9
```

Discussions: first of all, an array is declared to hold 10 integer numbers as : `int[ ] x = new int[10];` here, two things are done in one line, first, declares an array of x as integer data type and then next part is to declare spaces within physical RAM to hold ten integer numbers. Here, this point to be understood that 10 spaces are declared within memory for x and subscripts are from 0 to 9. So, the starting array is x[0] and last array is x[9]. To demonstrate the concept of handling arrays, here, how to insert value within array that is shown as:

```
x[i] = i;
```

When i=0, x[0]=0; and after that by the line I = i+1, I = 0+1=1, so, in 2nd iteration of loop, array value is x[1] = 1; in this way, last array value is x[9] = 9 and all these are printed. But, after printing the value of array x[9], value of 'i' increases by one and becomes 10, when it goes to loop control to check for while(i<=n) meaning while(10<=9) which is not satisfied and don't allow to enter within loop, finally it comes out of loop and program ends.

Arrays can take part in all mathematical operations and value of an array can be replaced by other values of other array also. Some examples are shown below:

```
x[i] = x[i] * 5;
x[i] = 6 - x[i];
x[2] = x[5] + x[3] * x[2] etc.
```

Problem: wap to declare an *int* array, start a while loop to input four numbers from keyboard, assign each number within the array and the print all numbers.

```
//ExArray.java : input and print 4 numbers using while() loop.
import java.util.*;
class ExArray
{
  public static void main(String args[])
  {
    //int x[ ];
    int[] x = new int[4];
    int i=0, n=4;
    Scanner keyboard = new Scanner(System.in);
    int number;
   //start a loop to continue upto i times
    while(i<=n)
    {
          System.out.print("Enter any number:   " );
          number = keyboard.nextInt();
          x[i]=number;
          System.out.println("The no is:"+  x[i]);
          i++; // i=i+1
    }
  }
}
OUTPUT:
Enter any number:   10
```

```
The no is:10
Enter any number:  20
The no is:20
Enter any number:  35
The no is:35
Enter any number:  31
The no is:31
Enter any number:  56
Exception in the method of "main" java.lang.ArrayIndexOutOfBoundsException:
Index 4 out of bounds for length 4 at ExArray.main(ExArray.java:17).
```

Discussion: in this example, four arrays are created as 'int[] x=new int[4];' so a loop should work for maximum four times to input/print any numbers within that array. But, in this example, carefully a loop has been crated as 'while(i<=n)' if we observe carefully will see in loop control section, i.e, i<=n, it will loop for upto n i.e, upto 4. It starts from zero and end at four as, i=0, i=1, i=2, i=3 and i=4; so we see 0 to 4 there are five numbers. We expected wrongly that x[4] exist, but, when we tried to input data within x[4] it gives the error message throwing exception message what we see at end of output. If we check output, will see four numbers are input and printed correctly, the fifth number it could not accept/hold because, we have not created any room to hold fifth number, so, this error message is given and program is terminated.

Problem: wap to define an one-dimensional array, assign some values within that and using a loop print values of the array.

```
//ArrayOne.java : declare one dimensional array, input data and print
import java.util.*;
class ArrayOne
{
  public static void main(String args[])
  {
    int i=2;
   int x[];
   //initialization of variables
   x= new int[3];//total 3 variables
   x[0]=5;
   x[1]=3;
   x[2]=45;
   for(int count=0;count<=i;count++)
   {
         System.out.println("value= " +x[count]);
   }//end of for
 }//end of main
}//end of class
OUTPUT:
value= 5
value= 3
value= 45
```

Discussions: working with arrays has three steps: declaration, initialization and assignment of values. Another important point must be carefully noted that if three arrays are initialized, then, first one is x[0], starts from zero, and last one is x[2], that is total-1. Arrays can take part in any mathematical operation as + - * / as a single variable. Some examples:

```
int result=x[1]+x[2]-x[0];
int result=x[0]*x[2]/x[1];
```

Also, any value can be reassigned within the components of the array individually as:

```
x[2]=56;
x[0]=x[2] + 50; etc.
```

Problem: wap to input some numbers within an integer array, then using for loop count number of even and odd numbers present within the array.

```
//deci1.java: example of one dimensional array: to count no of odd/even numbers
//present within an array using for loop.
//Logic is that if the number gives zero remainder after dividing by 2, then
// it is even number else it is odd number.
class deci1
{
	public static void main(String args[])
	{
		int n[]= {34, 25, 54, 28, 77, 74, 85, 82};
		int i=0, counter1=0, counter2=0;
		int high = n.length; // calculate length of array of n.
		for( i=0; i<high; i++)
		{
			if((n[i] % 2) == 0)
			{
				counter1 +=1; // counter1=counter1 +1
			}
			else
			{
				counter2 +=1;//counter2=counter2+1
			}
		}
		System.out.println("Even nos = " + counter1);
		System.out.println("Odd nos = " + counter2);
	}
}
OUTPUT:
Even nos = 5
Odd nos = 3
```

Discussions: in this example some new techniques are used, (i) assignment of values within an array as:

```
int n[]= {34, 25, 54, 28, 77, 74, 85, 82};
```

It is observed that in one line array is declared and values are inserted within the array also, but, how many numbers will be inserted or size of the array is not

declared before. JAVA can assign array size as per number of list. (ii) Another statement used to compute the size of the array is:

```
int high = n.length;
```

It is seen that there are 8 numbers within the array and the variable high will receive value 8. By this statement, an integer type of variable 'high' is declared which will contain the value of the length of the array n[].

Problem: assign some average marks in an integer array, then, based on logic of computing results of students show the division of result achieved by the students. Marks distributions are as:

```
=>80 STAR
>=60 and <80 FIRST DIV
>=45 and <60 2ND
>=33 and <45 3RD DIV
<33 FAIL
```

```
//deci3.java: Example of one dimensional array and for loop using if..else…else if
// decision making multiple structures for multiple decisions. In an array there are
//four marks, it is required to compute results based on set of conditions.
class deci3
{
	public static void main(String args[])
	{
		int mark[]= {65, 45, 88, 28};
		int i=0, counter1=0, counter2=0;
		int high = mark.length; // calculate length of array of n.
		for( i=0; i<high; i++)
		{
			if(mark[i] >=80)
			{
				System.out.println(mark[i] + " : Result is STAR");
			}
			else if(mark[i] >=60)
			{
				System.out.println(mark[i] + " : Result is FIRST DIV");
			}
			else if(mark[i]>= 45)
			{
				System.out.println(mark[i] + " : Result is 2ND DIV");
			}
			else if(mark[i]>=33)
			{
				 System.out.println(mark[i] + " : Result is 3RD DIV");
			}
			else
				System.out.println(mark[i] + " : Result is FAIL");
		}
	}
}
```

OUTPUT:
65 : Result is FIRST DIV
45 : Result is 2ND DIV
88 : Result is STAR
28 : Result is FAIL

Arrays of String data type:

Like numbers, if we need to input many string types of data, we can declare array of String data type. In JAVA, there is no end of input string, only it remembers where to start and where the string has ended. Within java package, String class is available under *'lang'* package as:
java.lang.String

Array of string is declared as:

```
String[] nm=new String[4];
```

Here, four arrays of 'nm' are created and also initialized. So, 'nm' is now ready to assign/input string type of data.

Problem: wap to declare a string type of array, using *Scanner* class input name and age of three students; use *while* loop for repeated input of name/age.

```
//ExArrayS.java: demonstration of integer and string types of arrays.
import java.util.*;
class ExArrayS
{
  public static void main(String args[])
  {
          int[] x;
          x=new int[4];
          String[] nm=new String[4];
          int i=0, n=3; // 4-1=3 , last subscript number.
          Scanner keyboard = new Scanner(System.in);
          Scanner snm=new Scanner(System.in);
          int number;
          //start a loop to continue upto i times
          while(i<=n)
          {
            System.out.print("Name ?");
             nm[i]=snm.nextLine();
             System.out.print(" Age  ?   " );
             number = keyboard.nextInt();
             x[i]=number;
             System.out.println("Name is :"+  nm[i]+" Age is: "+x[i]);
             i++; // i=i+1
          }
  }
}
```

```
E:\example>java ExArrayS
Name ?Ronit Das
 Age  ?  20
Name is :Ronit Das Age is: 20
Name ?Rahul Jana
 Age  ?  21
Name is :Rahul Jana Age is: 21
Name ?Arnab Sen
 Age  ?  22
Name is :Arnab Sen Age is: 22
Name ?Anjali Sharma
 Age  ?  20
Name is :Anjali Sharma Age is: 20
```

Discussion: in this the last subscript number is taken 4-1=3 which generates nm[0, nm[1], nm[2] and nm[3] these four arrays of ***String*** data type. But, it is noted that ***Scanner*** class has some limitations to handle inputs of multiple variables. More efficient class is, ***DataInputString*** and its method ***readLine()*** is the best way to input string type of data which will be discussed in later chapters.

Home Work: (1) edit this program as: input 4 names and 4 ages of students using *while* loop, after that develop another *while* loop to print name and age of all 4 students what were inputted.

Two Dimensional Arrays:

One dimensional array is required for inputting data of different units like: distance, weight, height, time etc. The advantage of this declarations of array is that by declaring one variable name we can hold long data as example: int[ ] x;
In some applications we need to input data as real life table or database having rows and columns. In any table, it has rows and columns. If we think like a graph sheet, it has X and Y axes and each of the point has subscripts like (x1,y1), (x2, y2) …. (xn, yn) where there are 'n' numbers of two dimensional points on the graph sheets. We can say these points are p1, p2…pn while each of the points are defined by two values of x and y. So, it is called two dimensional points. If we like to define different points within a cube or within a living room, then we need three values of (x,y and x) components as P(x,y,z); so, this system is of three dimensional points. In mathematics, it has another name called features, as example, in this case point P has three features and position of P is defined by three features like values of x,y and z. In computer programming, any object is defined by its features or characteristics, as example, color of an a ball is defined by R(red),G(green),B(blue) index values. A CPU box of a computer is physically defined by L(length),H(height), W(width), K(weight) etc. We can define a CPU box as CPU(L,H,W,H), it means it is a four dimensional object. In JAVA putting multiple [ ] symbols we declare multiple dimensional parameters.
For two dimensional points, if we want to work with computers, we need to declare points as two dimensional variables as:

```
int[ ][ ] point;
point = new int[][];
```

or, in one line it can be written as:

```
int[ ][ ] point = new int[x][y];
```

Now, if we need four points to declare, then we need four values of x and four values of y, so, we can write as: int x = 4, y=4;
So, the above declaration will look as :

```
int[ ][ ] point=new int[4][4];
```

In real life the variable is: point[x][y], if it is of three dimensions, it will look like: point[x][y][z].

Say, we are declaring x[2][3], then there are 2*3=6 variables or rooms are created to hold six numbers. Names of these variables are as:

x[2][3] =
x[0][0] , x[0][1] , x[0][2]
x[1][0] , x[1][1] , x[1][2]

N.B. numbering starts from zero.
So, six variables are created within the memory of computer. We can input data by assignment or by input from keyboard as shown below:

```
x[0][0] = 57;  x[1][2] = 4589 etc.
```

If we declare the variable as String data type, we can assign text data as:

```
String x[1][1]= "Rahim is a Good Boy";
```

Below, an example is shown to input from keyboard into this type of two dimensional array, when the array is filled by six numbers, then, all rooms of the array are filled. Now, we can used these values in any operations. Below, is the example of a nested for( ) loop for inputting data and another nested loop is used for displaying data. A nested loop is a loop structure within a loop. For handling of two dimensional data, we need one level nesting loop, i.e. only one loop within a loop. But for handling three dimensional data, two levels of nested loop are required. The concept is shown below:

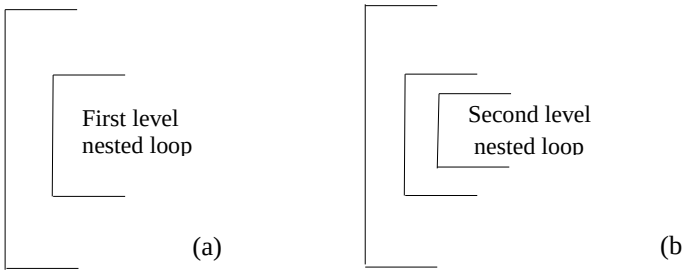

Fig. 3.2. different levels of Nested Loop. (a) two loops (b) three loops

```
//ArrayTwo.java : declare two dimensional array, input data and print
import java.util.*;
class ArrayTwo
{
  public static void main(String args[])
  {
    int row = 2, col = 3;
    int[ ][ ] x = new int[row][col];
    Scanner keyboard = new Scanner(System.in);
    int number;
   //start a loop to continue upto i times
    for(int r = 0; r < row; r++)
    {
          for(int c = 0 ; c < col ; c++)
          {
            System.out.print("Enter any number:   " );
            number = keyboard.nextInt();
            x[r][c]=number;
        }// col ends here
   }//row ends here
   //array is filled by 2*3=6 numbers
   //now display those numbers
   for(int r=0; r<row; r++)
    {
          System.out.print("Row- " + r +" is : ");
          for(int c=0; c<col;c++)
          {
            System.out.print(x[r][c] + ", ");
     }// col
          System.out.println("");
   }//row
 }
}

E:\example>java ArrayTwo
Enter any number:   1
Enter any number:   2
Enter any number:   3
Enter any number:   4
Enter any number:   5
Enter any number:   6
Row- 0is :1, 2, 3,
Row- 1is :4, 5, 6,
```

Discussions: This program demonstrate two dimensional arrays as x[r][c]; in first part there is one nested for() loop used for inputting six numbers. To display data, again nested loop is used. Users should remember sequences of input, because, that sequence is required to read data from x[r][c] array. If we look at the output, it is

seen, in first row there are three data as 1, 2, 3 and in 2nd row there are three data as 4, 5, 6. It is a table like setup of data.

Table like data arrangements:

All are familiar with tables. Here, we are giving an example of a table which contains data of consumption of calories in a week. For inputting only the calorie column and considering week-days column as fixed, we see, there are column=1 and row=7.

Mon	2400
Tues	2350
Wed	2200
Thurs	1800
Fri	2000
Sat	2100
Sun	2159

For writing program, we need to edit the previous program as:

```
for(int r = 0; r < 6; r++)
    {
          for(int c = 0 ; c < 1 ; c++)
          {
            System.out.print("Enter any number:   " );
            number = keyboard.nextInt();
            x[r][c]=number;
         }// col ends here
   }//row ends here
```

By this code, it will fill the arrays x[0][0], x[1][0], x[2][0], x[3][0],x[4][0], x[5][0] by six numbers assuming the first column of week-names are fixed. If we need to read data from the array and to display from x[r][c], same code as shown in previous program should be used.

Suppose, it is required to record temperatures of seven days(Monday to Sunday) of a month dates from 1 to 30. In that case, in column side names of weekdays and in row side number of days of the month to be recorded. The table looks like:

Table 3.1. an example of table showing 7 columns and 30 rows

	Mon	**Tues**	**Wed**	**Thurs**	**Fri**	**Sat**	**Sun**
1	12	13	12	15	16	13	13
2	13	14	12	11	15	16	15
3	14	15	13	16	17	16	14
…..	….	….	….	….	….	….	….
30	12	15	16	17	17	18	18

If careful observation is done, it will be seen that row is from 1 to 30 and column grown from 1 to 7. To write a program in input data like this table, the previous program ‘ArrayTwo.java’ should be edited for row and column as shown below:

```
for(int r = 0; r < 29; r++)
    {
          for(int c = 0 ; c < 6 ; c++)
          {
            System.out.print("Enter any number:   " );
            number = keyboard.nextInt();
            x[r][c]=number;
         }// col ends here
   }//row ends here
```

It is noted that 29=30-1 as it starts from 0, in the same way, column, 7-1=6. For printing data from the array, the same logic will be used as per previous example.

Matrix Operations:

Matrix is a table like structure which contains some numerical data. To look at it is a table but, how the data are arranged within each row and column that to be known. What are these data for? What is the meaning of these data? Then, it is possible to do some mathematical operation over these data. A matrix is written as RxC matrix meaning that R is number of rows and C is the number of columns present in the table. A Matrix is a two dimensional array having 2D subscript numbers for each elements of the matrix as: A(0,0), A(0,1), A(0,2) of a row of three data. As example, two simple matrices A (2x3) and B(2x3) are shown.

A =

10	15	17
6	11	13

B =

10	15	17
6	11	13

We can do simple matrix operations as A+B, A-B, A*B and A/B element by element. What type of mathematical operations can be done within two matrices that depends on arrangement of data between the two matrices.

Matrix does linear mathematical equation's solutions. If the power of the variables is one that is known as Linear Equation as example:

Ax + By + Cz=0 or 5x + 8y + 9z=0 or 12x + 5z=0 or 35y – 3z=10 etc.

Really, the data of a matrix is the coefficient values of a linear equation. If we look at Matrix A there are two rows and these have come from these two equations as example:

10x + 15y + 17z = 0
and
6x + 11y + 13z = 0

In the same way, we can imagine two different linear equations for matrix B also.

Problem: A milk supplier has packets of different types of milks of a packer of half litre having prices 25, 26 and 24. Rate of Matrix is as:

25
26
24

The supplier has supplied for Monday, Tuesday and Wednesday to three Regions as this Matrix known as quantity matrix:

Mon	Tues	Wed
5	8	7
3	6	12
16	13	4

Now, it is required to compute total sales amount for Region wise and then to compute total sales in all Regions.
Solution: it is shown that rate is a 3x1 matrix and qty is a 3x3 matrix. Each of the data within the matrix has subscript value as: rate(row,col) and qty(row,col). From the above figure, it is seen that the sales to Region1 is:
25*5 + 25*8 + 25*7 or in programming concept it is:

rate(0,0)*qty(0,0) + rate(0,0)*qty(0,1) + rate(0,0)*qty(0,2)
This concept of simple Matrix linear multiplication is shown in the program below:

```
//Matrix1.java : declare two dimensional array, input data and print
import java.util.*;
class Matrix1
{
  public static void main(String args[])
  {
    int row=3, col=3;
    int[][] qty=new int[row][col];
    int[] rate= new int[row];
    Scanner kb = new Scanner(System.in);
    int number, no2;
          //input rate Matrix
          Scanner rt = new Scanner(System.in);
          for(int r=0;r<row;r++)
          {
           System.out.print("Price-rate of region : "+r +" :");
            no2 = rt.nextInt();
            rate[r]=no2;
          }
    //input 2nd Matrix of 3x3
    System.out.print("\nInput data to Qty Matrix\n");
    for(int r=0;r<row;r++)
    {
          System.out.print("Enter value for Row: "+r );
          for(int c=0; c<col; c++)
          {
```

```
            System.out.print("\nEnter any number for column: " + c +" :" );
            number = kb.nextInt();
            qty[r][c]=number;
      }// col
    }//row
    //array is filled by 3*3=9 numbers
    //now display those numbers
    System.out.print("\n\n");
    for(int r=0; r<row; r++)
     {
          System.out.print("Elements of Row- " +r+" is :");
          for(int c=0;c<col;c++)
          {
            System.out.print(qty[r][c] + ", ");
      }// col
          System.out.println(" ");
    }//row
          System.out.print("\n");
          //to compute regional total for all Regions
          int region_total[]= new int[row+1];
          int sales_tot=0,cc, row_total=0, total_sales=0;
    for(int r=0;r<row;r++)
     {
          //System.out.print("Row- " +r+"is :");
          for(cc=0;cc<col;cc++)
          {
            row_total=row_total+qty[r][cc];
      }// col
          System.out.println("\nTotal quantity supplied to Region : "+r +" is= "+
row_total);
          System.out.println("Rate for region : "+ r+ " is ="+ rate[r]);
             region_total[r]=rate[r]*row_total;
             System.out.print("Region total Sales : "+ region_total[r]);
             row_total=0;
             total_sales=total_sales+region_total[r];
    }//row
          System.out.print("\nTotal sales of all Regions : "+ total_sales);
  }
}
```

```
C:\example>java Matrix1
Price-rate of region : 0 :25
Price-rate of region : 1 :27
Price-rate of region : 2 :24

Input data to Qty Matrix

Enter value for Row: 0
Enter any number for column: 0 :5

Enter any number for column: 1 :8
```

```
Enter any number for column: 2 :3
Enter value for Row: 1
Enter any number for column: 0 :5

Enter any number for column: 1 :9

Enter any number for column: 2 :1
Enter value for Row: 2
Enter any number for column: 0 :2

Enter any number for column: 1 :25

Enter any number for column: 2 :10

Elements of Row- 0 is :5, 8, 3,
Elements of Row- 1 is :5, 9, 1,
Elements of Row- 2 is :2, 25, 10,

Total quantity supplied to Region : 0 is= 16
Rate for region : 0 is =25
Region total Sales : 400
Total quantity supplied to Region : 1 is= 15
Rate for region : 1 is =27
Region total Sales : 405
Total quantity supplied to Region : 2 is= 37
Rate for region : 2 is =24
Region total Sales : 888
Total sales of all Regions : 1693
```

Discussions: This program has three sections, input data in two matrices, compute summations and display output. At first, rate matrix is inputted by 3 data, then qty matrix is inputted by 3x3=9 data. When both matrices are filled by data, then, all data of this matrix are displayed row wise. Then a nested loop of for() is used to add all quantities sold to Region1 for Mon+Tues+Wed, after that total quantities are multiplied by the rate of that region1 to get total amount of sales in Region1. This loop iterates for three times to compute total sales for Region1, Region2 and Region3. Finally these three subtotals are summed to get net total sales.

Assignments; wap editing the above program where rate matrix will be a 3x3 matrix showing that rates of milk varies for Mon, Tues, Wed. Finally compute region wise sub-total and net total amount of sales.

Conclusion: array concept is discussed in details with the help of practical programs. Arrays can be one, two, three, …, n dimensional. One dimensional array is like a row of a table and two dimensional array is like a table of a spread sheet. It is explained that each element of data has an address which is represented by subscript value like: salary[r] or qty[r,c] etc. with the help of program of loop, any element of data can be accessed. Manually, it is not possible if the data table is very large, only programming concept of array and loop can solve this problem.

Link of PPT of this book and all source codes:

https://drive.google.com/drive/folders/1bMxMCaqPe0W35COAdn_-BrnV_uzQ-tNO?usp=sharing

(Note: to get access, copy and paste this link to your browser)

N.B. We appreciate readers of this Book, PDF presentation and users of all java programs stored in repository, for sending us errors if found any to the address dulal@actsoft.org.